La algorítmica distribuida

Roberto Gómez Cárdenas
rogomez@itesm.mx
http://webdia.cem.itesm.mx/ac/rogomez

La algoritmica

- Diseño algoritmos constituye una parte fundamental de las ciencias computacionales.
- Algoritmos estudiados en un contexto de programación secuencial
  - control de flujo del algoritmo es único y en cada paso del cálculo se realiza una acción única.
  - se estudia el manejo de estructuras de datos (arboles, listas, matrices, grafos, etc) y el uso de estas estructuras para producir un cálculo final.

Algoritmica paralela

- Algoritmica secuencial largamente estudiada.
- No es el caso para la algoritmica en un contexto paralelo
  - varios flujos de control pueden coexistir simultaneamente
- En un principio paralelismo y su algoritmica es estudiado dentro del área de sistemas operativos
  - el diseño del sistema esta dividido en actividades elementales de cálculo (procesos), de los que hay que administrar las interacciones

Los algoritmos y las redes

- Fenomeno redes, junto con el progreso de la metodología de programación introdujeron el concepto de comunicación como base del diseño de algortimos paralelos
- Nace una nueva algoritmica basada en el intercambio de mensajes entre actividades.
- Estos algoritmos paralelos son denominados protocolos.

Los protocolos

- Realizan servicios de transferencia de información o de control de actividades, dentro un conjunto de procesos que:
  - constituyen una aplicación
  - se ejecutan sobre distintos sitios
  - son enlazados por vías de comunicación
- Dentro del contexto de sistemas distribuidos estos algoritmos se conocen como *algoritmos distribuidos.*

Computo local vs distribuido

- El termino computo local se refiere a los programas que están confinados a un solo espacio de direcciones.
- El termino de computo distribuido se refiere a programas que hacen llamadas a otros espacios de direcciones, posiblemente en otra máquina.

¿Qué es un sistema distribuido?

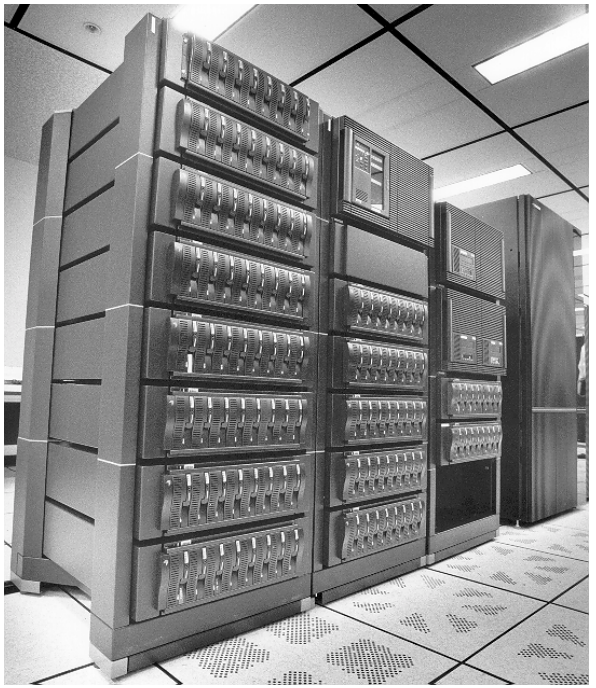
- Conjunto de entidades que se comunican entre ellas a través de mensajes, los cuales son enviados sobre vías de comunicación.
- Otras definiciones

> "A distributed system is one in which the failure of a computer you didn't even know existed can render your own computer unusuable"
> *Leslie Lamport*

> "A distributed system is one that stops from getting any work done when a machine you've never never heard of crashes"
> *Distributed Systems (Ed. Sape Mullender)*
> *edition 1, ACM Press 1989*

Tipos sistemas distribuidos

- Redes de computadoras
- Computadoras multiprocesadores
- Procesos cooperantes
- Celdas de manufactura
- Redes inalámbricas
- Computación móvil



Definición algoritmos distribuidos

- Abstracción lógica de un sistema distribuido, se habla de un conjunto de procesos y de líneas de comunicación virtuales
- Conjunto de procesos que, comunicandose a través de mensajes, llevan a cabo una tarea en común.

algoritmo distribuido = procesos + mensajes

Características algoritmos distribuidos

- Desconocimiento del número de procesos
- Desconocimiento del estado global
  - algoritmo centralizado puede tomar decisiones basados en el estado del sistema
  - nodos en un sistema distribuido solo tienen acceso a su estado local y no al estado global del sistema entero
- Entradas independientes en sitios diferentes
  - proceso P_1 recibe entradas diferentes a proceso P_2
- Varias programas ejecutandose al mismo tiempo, empezando en tiempos diferentes y operando a diferentes velocidades

Características algoritmos distribuidos

- No determinismo en el procesador (processor nondeterminism)
  - un programa centralizado puede describir su cálculo de acuerdo a su entrada, dado un programa y su entrada sólo un cálculo es posible
  - la ejecución de un sistema distribuido es, generalmente, no deterministica, debido a las posibles diferencias en la velocidad de ejecución de los componentes del sistema

Características algoritmos distribuidos

- Falta de un sistema de tiempo global
  - sistema centralizado, los eventos estan totalmente ordenados de forma natural, para cada par de eventos uno ocurre antes o después de otro
  - la relación temporal de los eventos que constituyen la ejecución de un algoritmo distribuido no es total, dado dos eventos no se sabe cual ocurre antes o después
- Tiempos entrega de mensajes diferentes
  - un proceso emite m_1 a dos procesos, dependiendo de la ruta m_1 puede llegar en tiempo diferente a los procesos
- Orden entrega de mensajes desconocido
- Fallas en la comunicación y en los procesos

Elementos Algoritmos Distribuidos

- Procesos
  - reciben, manipulan, transforman y emiten datos

- Vías de comunicación
  - medio sobre el cual circulan los datos y que forman una red local dotado de propiedades estructurales y dinámicas.

Metodo comunicación entre procesos

- Memoria compartida
  - sincronia en base a variables compartidas
- Mensajes punto a punto y/o broadcast
  - sincronia por envio de mensajes
- Ejecución de procesos remotos (RPC)
  - RPC proporciona la base de la sincronia

Atributos diferencian algoritmos distribuidos

Caracteristicas propias de los algoritmos distribuidos que permiten evaluar y comparar diferentes algoritmos que llevan a cabo la misma función.

- Método de comunicación entre procesos
- El modelo del tiempo
- El modelo de fallas
- El grado de distribución
- Estado global o local
- Los problemas a los que están dirigidos
- Complejidad (tráfico generado)

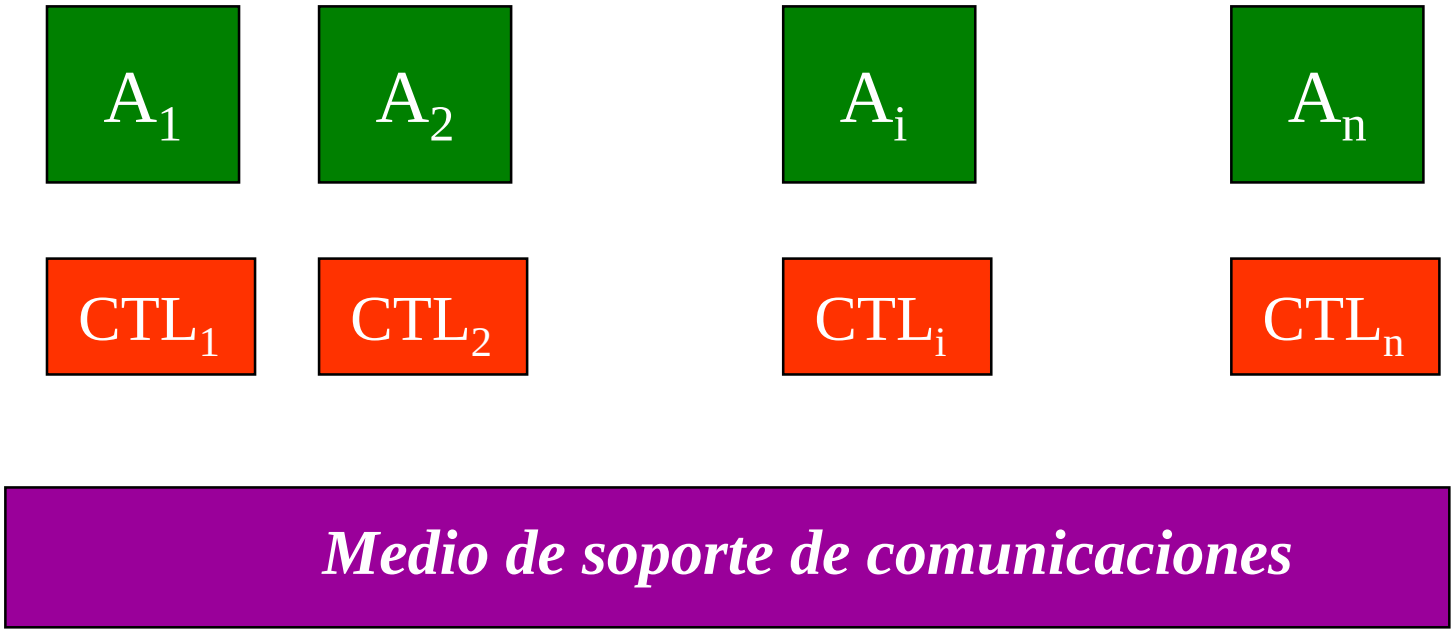
Los problemas a los que están dirigidos

- Problemas de aplicación
  - Son más herramientas que algoritmos y definen una aplicación
  - Representan interfaz final entre usuarios y sistema distribuido
  - Se apoyan en arquitecturas de software como:
    - CORBA: Common Object Request Broker Architecture
    - COM: Component Object Model
    - EJB: Enterprise JavaBeans
- Problemas de control
  - Algoritmos dedicados a llevar a cabo una función de control.
  - La palabra control es diferente a la de cálculo.
    - no calculan un resultado el cual, una vez obtenido, termina con los procesos
  - Prestan un servicio y lo pueden prestar de nuevo, en todo momento que sea necesario.

Algoritmos distribuidos control

- Están por abajo de las aplicaciones
- Proporcionan dos tipos de servicios
  - Proveedor de primitivas
    - exclusión mutua
    - envío/recepción mensajes
    - control de concurrencia
    - administración de archivos
  - Observadores de propiedades
    - interbloqueo
    - terminación de la ejecución
    - recolectores de basura

Algoritmos de aplicación y control



CTL_i :control de la i-ésima aplicación

A_i: aplicación

Metodos diseño algoritmos distribuidos

- Existen diferentes metodos para diseñar y construir un algoritmo distribuido.
- Se siguen dos metodos:
  1. a partir de ciertas propiedades de invariancia que caracterizan el problema planteado, se definen procesos de tal forma que en cada etapa de la construcción del algoritmo se conservan las invariantes
  2. el otro es de naturaleza empírica y consiste en construir un algoritmo y después verificar que soluciona el problema planteado

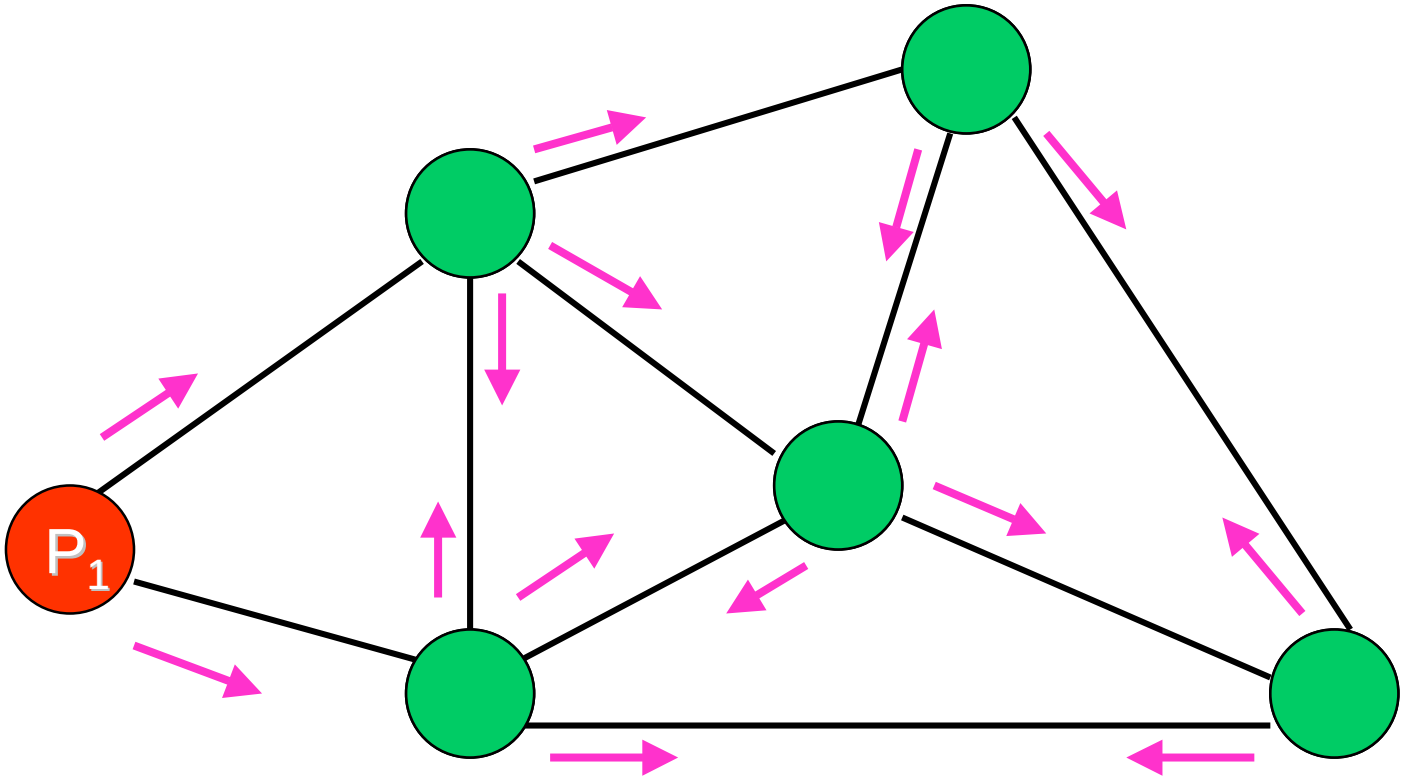
Técnicas diseño algoritmos distribuidos

- Independientemente de la metodología usada, la algorítmica distribuida usa técnicas clásicas que se encuentran en las redes.
  - acuse recepción mensaje, broadcast a un conjunto de procesos, etc
- Existen conceptos propios a la algorítmica distribuida
  - la difusión de cálculo
  - la ficha circulante
  - el estampado

La difusión del cálculo

- Propuesto por Dikstra y Schloten en 1980
- Procesos conectados de forma arbitraria por las vías de comunicación.
- Inicialmente un proceso emite mensajes a sus vecinos.
- El resto de los procesos solo puede emitir mensajes si ha recibido uno.
- Algunos algoritmos distribuidos realizan un control (terminasión, interbloqueo) utilizan este principio sobre un árbol de recubrimiento.

Ejemplo difusión cálculo



La ficha circulante

- Se hace circular un privilegio dentro de un conjunto de procesos conectados a través de una topologia de anillo.
- El anillo puede ser definido estáticamente o se puede reconfigurar dinámicamente.
- Técnica base de algoritmos distribuidos como los de exclusión mutua y de terminación.

La estampilla

- Basado en el concepto de relojes lógicos de Lamport
- Utilizado en diferentes algoritmos distribuidos en los que las elecciones deben ser equiparables
  - algoritmos de exclusión mutua y de interbloqueo
- En caso de conflictos el proceso no debe ser el mismo para evitar situaciones de hambruna.
  - sistemas centralizados utilizan un mecanismo de acceso a la memoria central o a un reloj global
  - en los algoritmos distribuidos estos dispositivos centralizados no existen
  - Lamport propone un orden total de eventos basados en el concepto de estampilla de tiempo

Los relojes lógicos

- ¿Qué fue primero, el huevo o la gallina?
- Propuestos por Leslie Lamport

  Times, cloks and ordering of events in a Distributing System,
  Leslie Lamport, Comm. ACM, Vol. 21, No. 7, Julio 1978

- Basados en la relación paso antes

Algunos algoritmos distribuidos de control

- Algoritmos de exclusión mutua
- Algoritmos de elección
- Algoritmos de ruteo
- Algoritmos de detección de terminación
- Algoritmos de detección de interbloqueo
- Algoritmos de cobertura de árbol (spanning-tree)
- Algoritmos de recorrido de gráfos
- Algoritmos de flujo maximal

La exclusión mutua

- Exclusión mutua: asegurar que sólo un proceso tiene acceso a un recurso compartido por varios procesos en un momento dado
- Sección crítica: parte del código donde el proceso utiliza el recurso compartido
- Procesos no comparten memoria en común
- Procesos se comunican a través de mensajes

Algoritmos Exclusión Mutua

- El algoritmo de LeLann
- El algoritmo de LeLann tolerante a fallas
- El algoritmo de Ricart y Agrawala
- El algoritmo de Misra

El algoritmo de LeLann

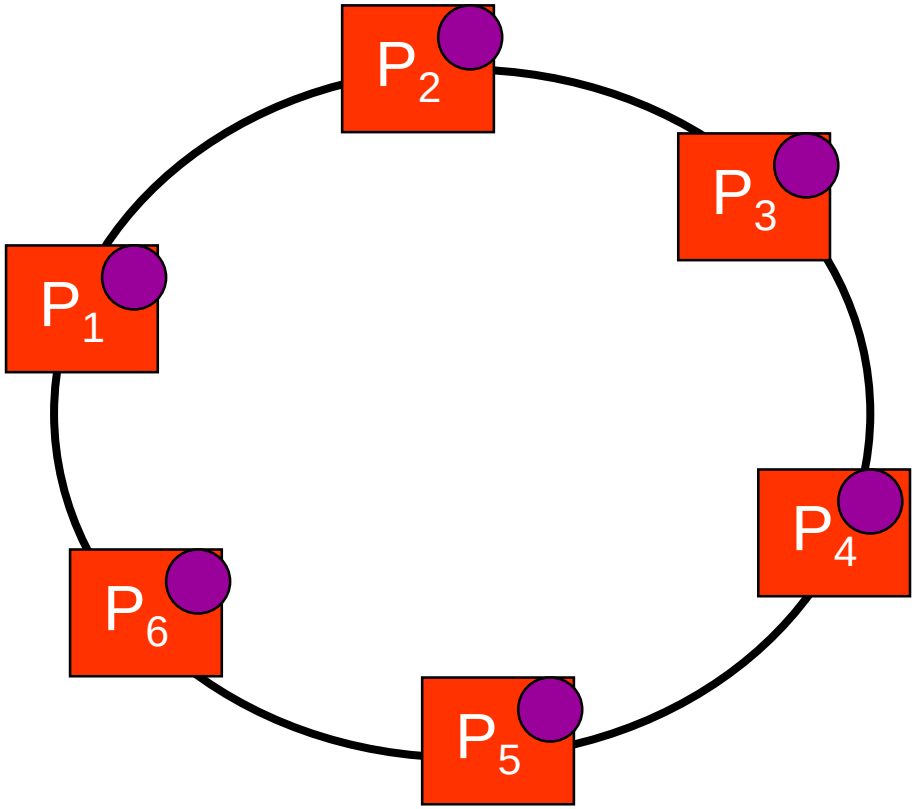
- Se hace circular una ficha
- El proceso que tiene la ficha esta en S.C.
- El algoritmo es el siguiente

esperar [ficha] de Pi-1

<Código Sección Critica>

enviar [ficha] a Pi+1

Ejemplo ejecución algoritmo



Algoritmos de elección

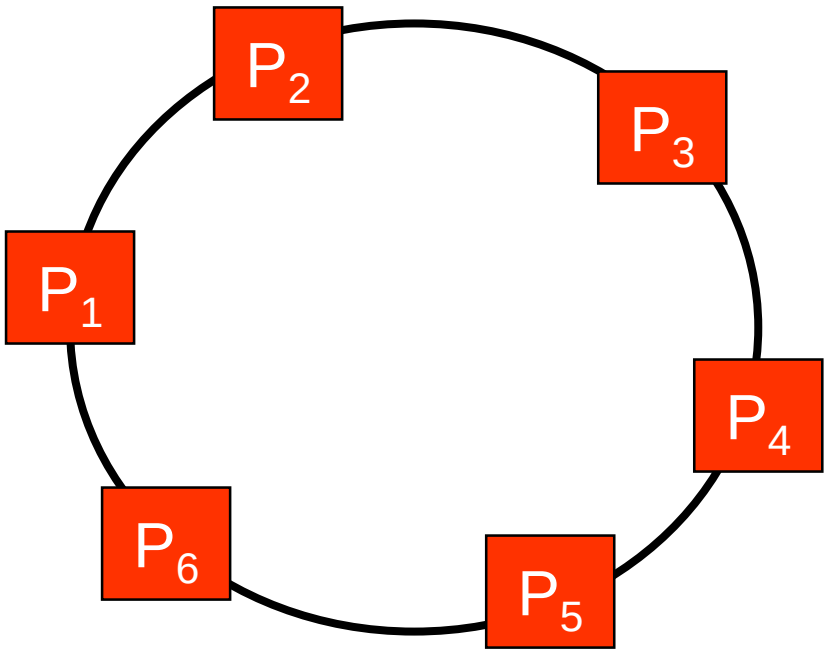
- En algunos algoritmos es necesario un proceso coordinador
- Si el proceso coordinador cae, es necesario que los procesos que queden elijan a uno de entre ellos
- Hipótesis general: cada proceso posee una identidad distinta de los otros, la cual usualmente es representada por un número
- La elección se realiza de acuerdo a dicha identidad

Ejemplos Algoritmos elección

- Algoritmo de LeLann
- Algoritmo de Chang & Roberts
- Algoritmo de Hirschberg y Sinclair
- Algoritmo Bulley (abusador)
- Algoritmo de Dolev, Klawe y Rodeh

Algoritmo Chang & Roberts

- Procesos agrupados en anillo unidireccional
- Cada proceso tiene un identificador único
- Se conoce el número total de procesos



Principo del algoritmo

- Cada proceso Pi envía su número a su vecino izquierdo Pj
- Cuando Pj recibe el mensaje compara el número con el suyo y envía el mejor a su vecino izquierdo
- Proceso que recibe mensaje con su número sabe que su número ha dado la vuelta y es el elegido

Elementos del algoritmo

- Variables de los procesos
- Tipos de mensajes
- Primitivas de comunicación

Variables de los procesos

- ego id_proc;
  - identificador del proceso
- participa booleano;
  - inicializado en falso
- coordinador id_proc;
  - identificador del ganador

Tipos de mensajes

- Mensaje 1: [elección, x]
  - eleccion: mensaje que incluye el identificador de un candidato
  - x es candidato a la elección
- Mensaje 2: [elegido, x]
  - elegido: mensaje que incluye al ganador
  - x ganó la elección

Primitivas de comunicación

- Solo se maneja una primitiva
  - env-vi[x]
  - envía mensaje x al vecino izquierdo del proceso emisor

Código del algoritmo

- Inicio del algoritmo
  - alguien decide realizar una elección
- Recepción del mensaje
  - que hacer cuando llega un mensaje
- Notificación del ganador
  - notificar al resto de los procesos que el algoritmo termino y quien gano

Inicio del algoritmo

```
decisión provocar una elección
        participa := verdadero;
        env-vi [elección, ego]
fin-decisión
```

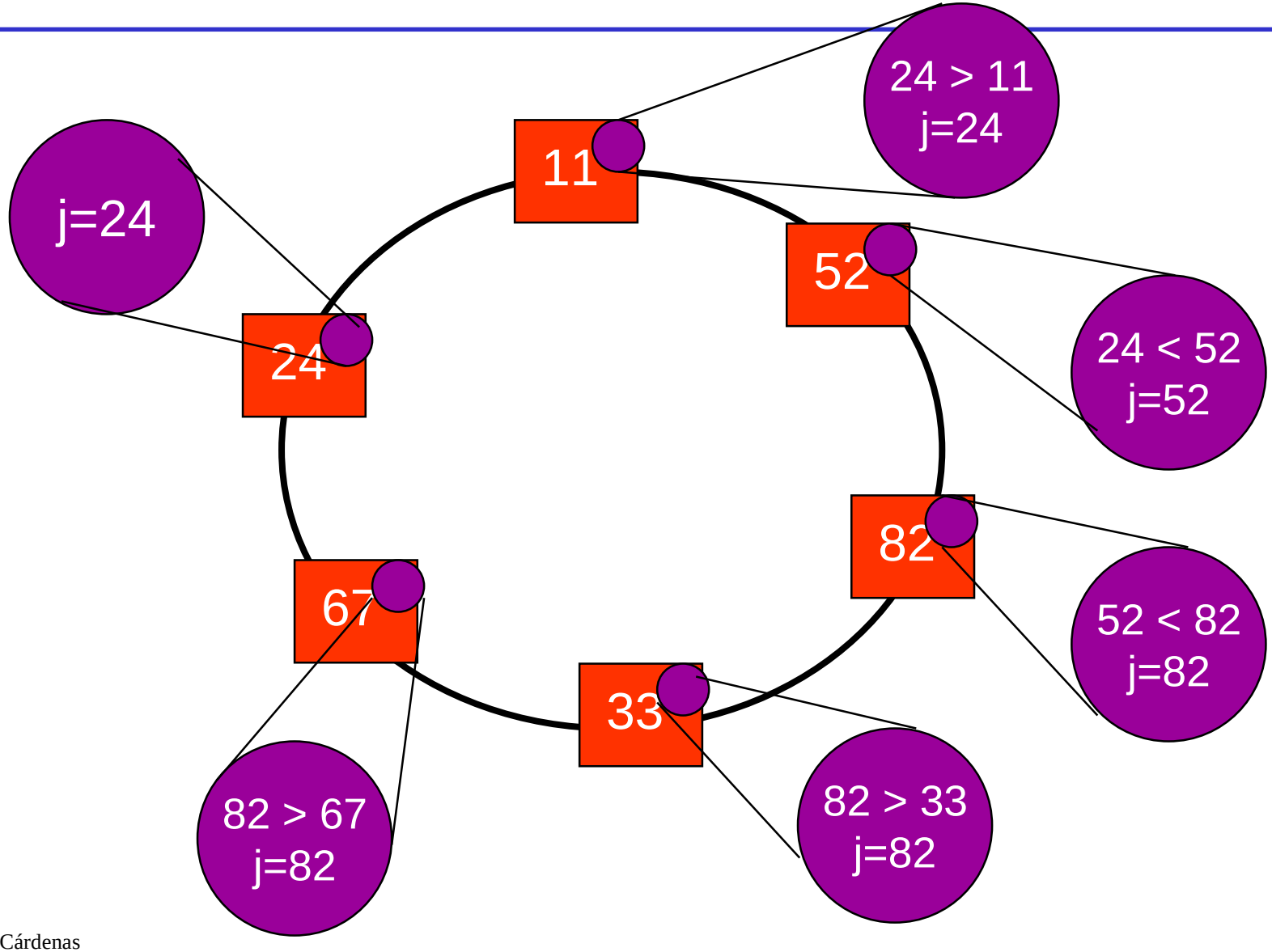
Recepción del mensaje

```
recepción_mensaje [elección, j]
    caso
       si (j > ego) entonces
          env-vi [elección, j]
          participa := verdadero
       si (j = ego) entonces
          env-vi [elección, j]
       si (j < ego) y (~ participa) ) entonces
          env-vi [elección, ego];
          participa := verdadero;
    fin-caso
fin-recepción_mensaje
```

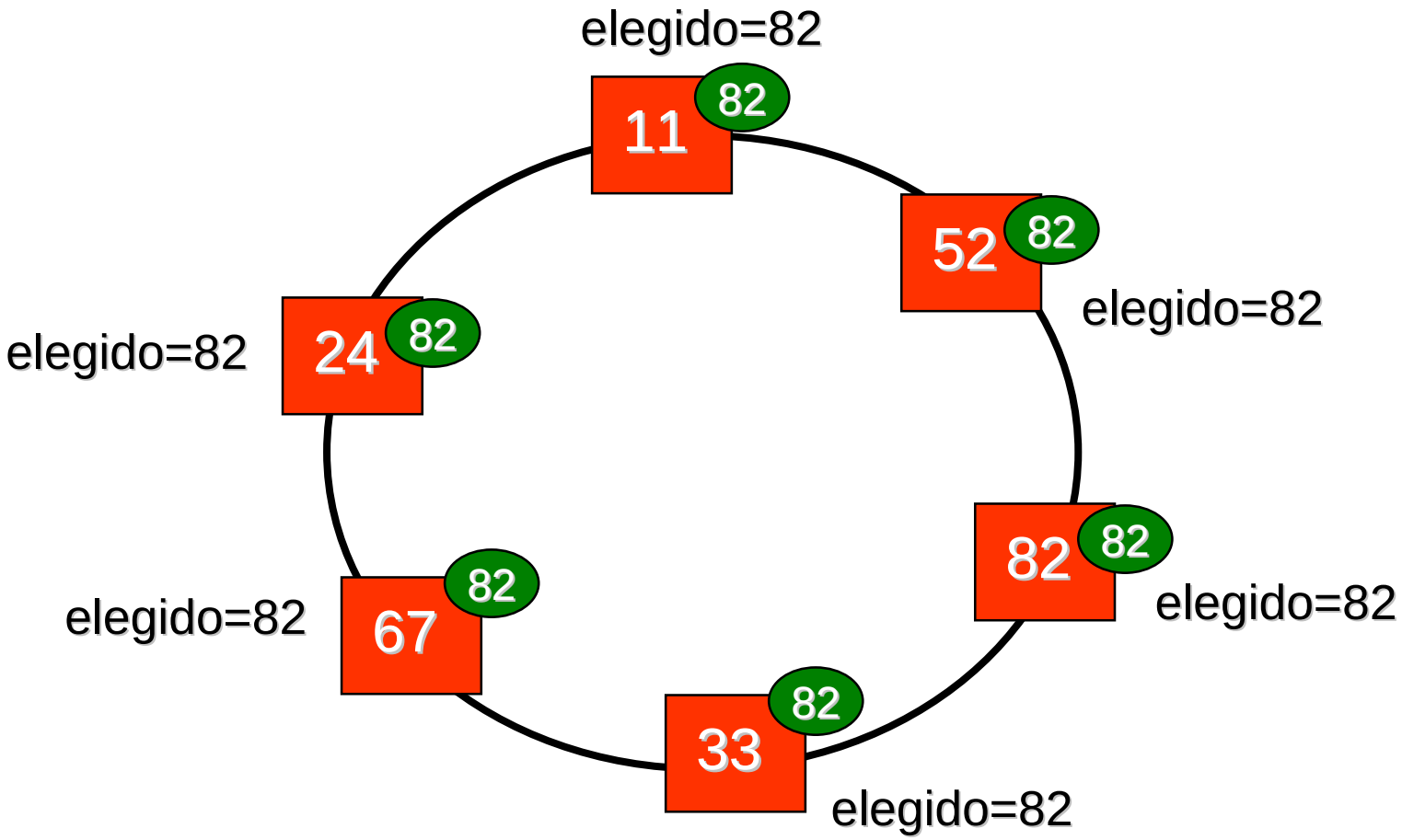
Notificación del ganador

```
recepción_mensaje [elegido, j]
        coordinador:= j;
        participante := falso;
        si (j    ego) entonces
                env-vi [elegido, j];
        fsi
fin recepcion_mensaje;
```

Ejemplo ejecución algoritmo



Ejemplo ejecución algoritmo



Algunos temas relacionados

- La ola recursiva
- Los algoritmos autoestabilizantes
- GRID COMPUTING
- AD HOC COMPUTING

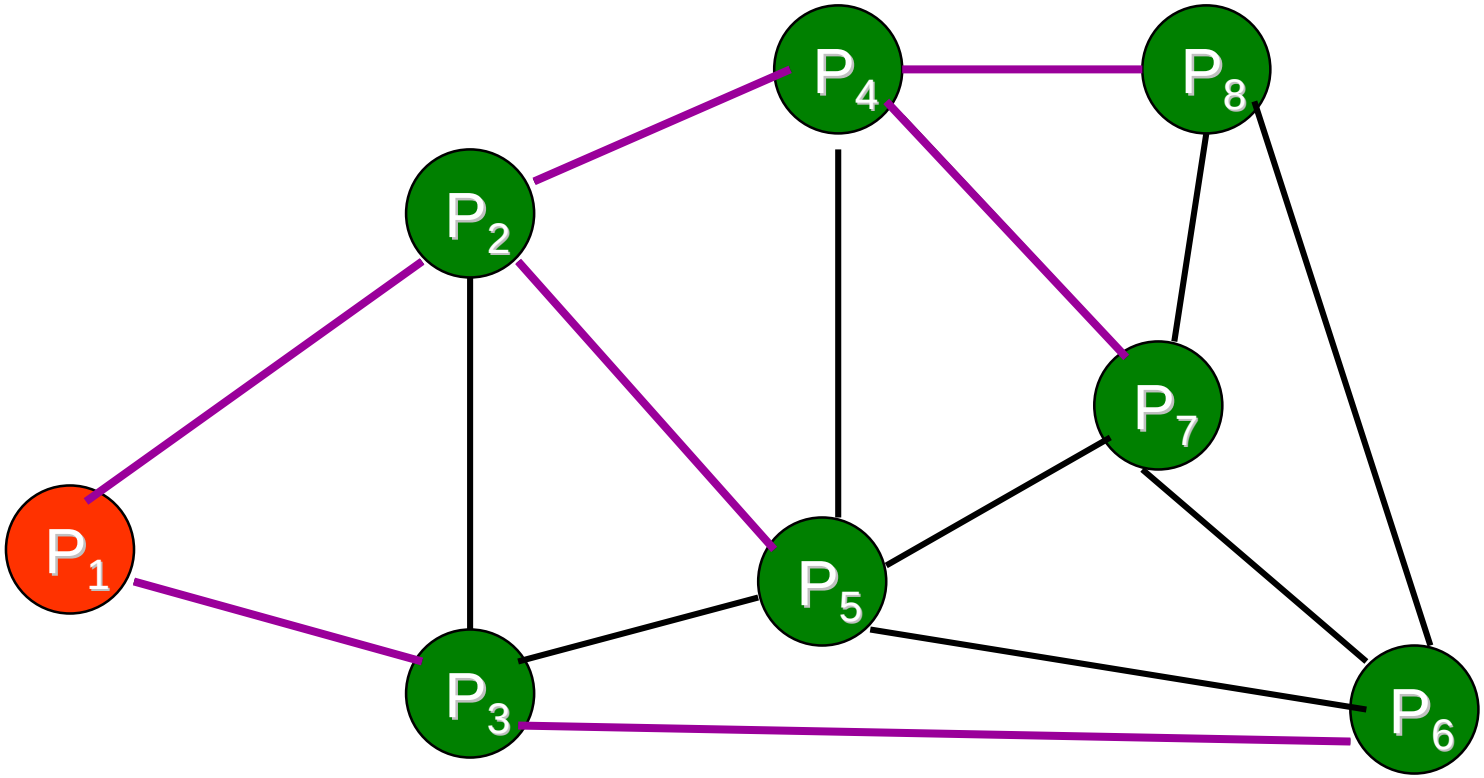
La Ola Recursiva

- Combinación de:
  1. modo de comunicación multipunto
     - ✓ difusión sobre un grupo
     - ✓ RPC sobre un grupo
  2. del principio de recursión
- Algoritmos tipo ola
- Definición de una *ola recursiva*

Características ola recursiva

- Estructura de *control distribuida*
- Procedimiento distante
- llamada *paralela* de n (o ninguna) ejecución de él mismo
- Ejecución induce un árbol
  - raíz asociada a la primera ejecución del procedimiento
  - raíz y nodos asociados a procedimientos en espera de que terminen las llamadas que realizaron
  - las hojas estan asociadas a: procedimientos activos ejecución final sin llamada recursiva

El árbol de la Ola Distribuida



Las primitivas utilizadas

- La instrucción *par*

  par i **in** <dominio> **do**

  <Bloque de instrucciones>

  enddo

- Sintaxis del RPC

  <nom_proc> **(**<lista parámetros>**) on** <ident_procs>

Tipos de ola recursiva

1. Ola recursiva secuencial
   - cada llamada genera solamente una sola llamada recursiva
   - la ola se propaga sobre una lista o anillo
2. Ola recursiva sobre un árbol
   - reposa sobre una topología de árbol definida anteriormente
   - cada nodo conoce a su padre e hijos
   - la raíz representa al proceso iniciador
   - hojas representan procesos encargados de detener la ola
3. Ola recursiva inundante
   - construcción dinámica de la estructura de control
   - uso de una estrategia de inundación

Esquema general de la ola

```
type id_processus is .....;
procedure ola_recursiva (<parámetros>) is
  <Declaraciones>;
  i: id_processus;
  procs_group: setof id_processus;
  begin
      <Bloque A de instrucciones>
      if (condición) then
          < Bloque B de instrucciones>
          par i in procs_group do
              < Bloque C de instrucciones>
              ola_recursiva_( f(<parámetros>) )on i;
              < Bloque D de instrucciones>
          enddo;
          < Bloque E de instrucciones>
      endif;
      < Bloque F de instrucciones>
  end ola_recursiva;
```

Ejemplo ola recursiva

```
procedure difunde ( msj ) is
   hijos: setof id_processus;
   info: <información a recibir/enviar>;
   i: id_processus;
begin
   info:=trata_info(msj);
   if not empty(hijos) then
        par i in hijos do
                difunde( info ) on i;
        enddo;
   endif;
end difunde;
```

Grid computing

- *Grid*, en español malla,
  - es adoptado del ámbito de la electricidad.
- Comparación: la electricidad es generada en algún lugar el cual no es indispensable saber su ubicación.
- En la computación el término *grid* implica la idea de acceder a recursos computacionales (v.g. supercomputadoras, clusters, sistemas de almacenamiento, etc) sobre una red, como Internet o una simple Intranet, sin tomar en cuenta los problemas de bajo nivel de saber en que lugar se encuentran,
  - además de presentar al conjunto de recursos como uno solo unificando los recursos para resolver problemas de gran escala

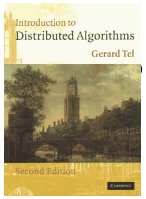
Ad-hoc computing

- El término latín ad hoc significa “a la medida”, “para un propósito en particular”.
- Colección de dos o más dispositivos (v.g. palmtop, laptop, teléfonos celulares, etc) equipados de una comunicación inalámbrica y con capacidad de interconectarse unos con otros, a través de ondas de radio
- Los nodos de éste tipo de sistema pueden detectar la presencia de otros nodos y realizar lo necesario para poder agregarse a la red y así poder compartir sus recursos y acceder a otros.

La autoestabilización

- En 1973, Dijkstra utilizó el término “autoestabilizante” para distinguir cualquier sistema que tiene la siguiente propiedad:
  - si el sistema empieza en cualquier, posiblemente estado ilegítimo, está garantizado a converger a un estado legítimo en un tiempo finito

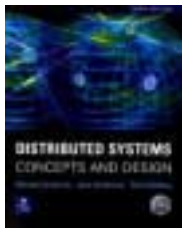
Referencias (libros)



Introduction to Distributed Algorithms, Gerard Tel, Cambridge University Press , 1994

Algorithmes Distribues & Protocoles, Michel Raynal, Ed. Eyrolles, 1985

Distributed Operating Systems & Algorithms, Randy Chow y Theodore Johnson, Ed. Addison Wesley, 1997



Distributed Systems, Concepts and Design , George Coulouris, Jean Dollimore, Tim Kindberg, Segunda Edición, Ed. Addison Wesley, 1994

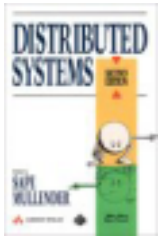
Referencias libros

Distributed Algorithms, Nancy A. Lynch, 1996, Morgan Kaufmann Publishers

Distributed Computing, Hagit Attiya and Jeeniifer Welch, Mac Graw Hill, 1998

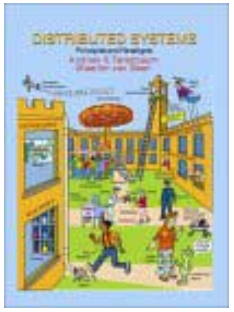
An introduction to distributed algorithms, Barbosa, V.C.,MIT Press, 1996,

Distributed Systems, S. Mullender, Ed. Addison Wesley



Algorithmique Paralléle et Distribuée, Ivan Lavallée, Primera Edición, París 1995, Ed. Hermes

y otros libros más



- Evaluation des algorithmes distribués, Christian Lavault, Ed. Hermes, 1995
- Algorithmique du parallélisme, Michel Raynal, Ed. Dunod, 1984
  - Algorithms for Mutual Exclusion, MIT Press, Cambridge, Massachusetts, 1986
- Synchronisation et controle des systems et des programmes repartis, M. Raynal, J.M. Helary, 1988
- Distributed Systems, A. Tanenbaum, Prentice Hall, 2002

Algunas sitios interesantes

- Distributed Algorithms and Systems
  - http://www.cs.chalmers.se/~tsigas/DISAS
- MIT Theory of Distributed Systems Group
  - http://theory.lcs.mit.edu/tds/
- Laboratorio de Investigación Avanzada de la Univ. Paris 8
  - http://lria.univ-paris8.fr/
- Centro de Estudios y de Investigación en Informática (CEDRI-CNAM)
  - http://cedric.cnam.fr/

Los principales congresos

- PODC (2004 – 23)
  - ACM Symposium on Principles of Distributed Computing
- OPODIS (2004 – 8)
  - International Conference On Principles Of DIstributed Systems

- DISC (antes WDAGS, 2004 - 18)
  - Symposium on DIStributed Computing

- ICDCS (2004 – 23)
  - International Conference on Distributed Computing Systems
- SIROCCO (2004 – 11)
  - International Colloquium on Structural Information and Communication Complexity

Otros eventos relacionados

- SPAA (2002 – 14)
  - ACM Symposium on Parallel Algorithms and Architectures
- ISADS
  - International Symposium on Autonomous Decentralized Systems
- International Summer School On Distributed Algorithms (2002 – 7)
  - University of Siena, Isola d'Elba, Livorno, ITALY